HY220: Εργαστήριο Ψηφιακών Κυκλωμάτων

Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης
Εαρινό Εξάμηνο 2026

Εργαστήριο 1 *(2 εβδομάδες)*

- **Εβδομάδα 1** (αναλόγως το εργαστηριακό τμήμα που έχετε δηλώσει)
  (Γκρουπ Α) Τρίτη 21/04 13:00 – 15:00 στην αίθουσα Β.110
  (Γκρουπ Β) Παρασκευή 24/04 11:00 – 13:00 στην αίθουσα Β.110

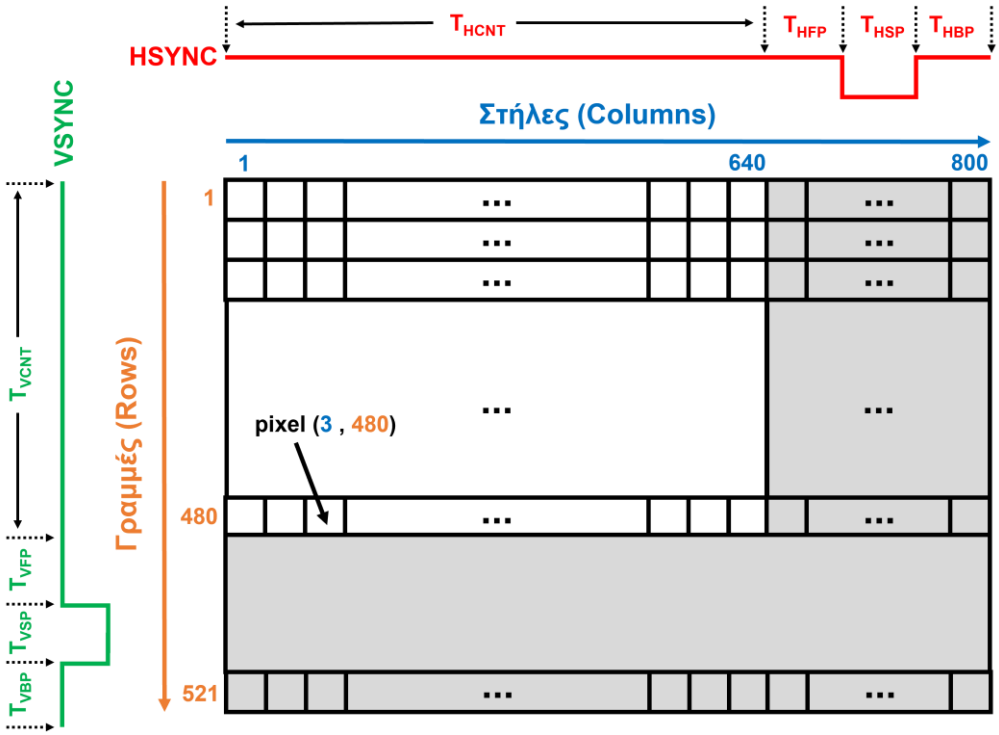
- **Εβδομάδα 2** (αναλόγως το εργαστηριακό τμήμα που έχετε δηλώσει)
  (Γκρουπ Α) Τρίτη 28/04 13:00 – 15:00 στην αίθουσα Β.110
  (Γκρουπ Β) Τετάρτη 29/04 13:00 – 15:00 στην αίθουσα Β.110
  ~~**(Γκρουπ Β)** Παρασκευή 01/05 11:00 – 13:00 στην αίθουσα Β.110~~

Κατά τη διάρκεια των εργαστηρίων θα υλοποιήσετε σε τρεις φάσεις το παιχνίδι «Λαβύρινθος» (Maze). Περιληπτικά, το τελικό παιχνίδι θα εμφανίζει σε VGA οθόνη ένα λαβύρινθο και τη φιγούρα ενός παίκτη. Ο χρήστης θα μπορεί να μετακινήσει τον παίκτη με τα κουμπιά που υπάρχουν πάνω στην πλακέτα και ο σκοπός είναι να τον οδηγήσει στην έξοδο το ταχύτερο δυνατόν.

Στο *Εργαστήριο 1* θα υλοποιήσετε την πρώτη φάση (**σε 2 εβδομάδες εργαστηρίων**) που περιλαμβάνει την οδήγηση μιας οθόνης VGA για την εμφάνιση οριζόντιων γραμμών με διαφορετικά χρώματα. Για το εργαστήριο αυτό θα σας δοθεί ο σκελετός αρκετών κομματιών σε SystemVerilog, ένα κατάλληλο testbench, reference outputs και ένας προσομοιωτής VGA (VGA Simulator) που σας δίνει τη δυνατότητα να βλέπετε τι θα εμφανίζονταν σε μια πραγματική οθόνη από τον κώδικά σας.

Χρονισμός VGA 640 x 480 @ 60Hz

Ένα πλήρες frame (εικόνα) αποτελείται από pixels που το καθένα περιέχει τιμές για τα χρώματά του σε RGB (red/green/blue). Το frame αποτελείται από στήλες (columns) και γραμμές (rows). Για κάθε γραμμή πρέπει να δίνονται τα pixels της κάθε στήλης σε διαδοχικούς κύκλους ρολογιού και μόλις ολοκληρωθεί ο απαιτούμενος αριθμός από pixels (columns) τότε πρέπει να σηματοδοτηθεί το τέλος της γραμμής έτσι ώστε να ξεκινήσει η επόμενη γραμμή (N+1) από την αρχή στην στήλη 1. Η σηματοδοσία για το τέλος της γραμμής γίνεται με τον παλμό HSYNC (horizontal synchronization – οριζόντιος συγχρονισμός). Όταν τελειώσουν όλες οι γραμμές τότε πρέπει να σηματοδοτηθεί το τέλος της εικόνας (frame) έτσι ώστε να ξεκινήσει το επόμενο frame από την αρχή στη γραμμή 1 και τη στήλη 1. Η σηματοδοσία για το τέλος του frame γίνεται με τον παλμό VSYNC (vertical synchronization – κατακόρυφος συγχρονισμός). Στην παρακάτω εικόνα φαίνεται το παράδειγμα ενός frame 640 x 480 (δηλαδή 640 στήλες και 480 γραμμές).



Για κάθε γραμμή απαιτείται χρόνος T_{HCNT} για τα ενεργά pixels που θέλουμε να εμφανίσουμε και μετά πρέπει να ακολουθήσει ο παλμός HSYNC (με κόκκινο στην εικόνα) ο οποίος είναι active-low για χρονισμό VGA 640 x 480. Για τη δημιουργία του παλμού HSYNC (δηλαδή μετά τα ενεργά pixels T_{HCNT}) απαιτείται κάποιος κενός χρόνος πριν τον παλμό T_{HFP} (horizontal front porch), κατά τη διάρκεια του παλμού T_{HSP} (horizontal sync pulse) και μετά τον παλμό T_{HBP} (horizontal back porch). Σε όλη αυτή τη διάρκεια τα pixels είναι «κενά» (πρέπει να έχουν τιμή 0), δεν λαμβάνονται υπόψιν και δεν εμφανίζονται (με γκρι στο δεξί μέρος της εικόνας).

Για κάθε εικόνα (frame) απαιτείται χρόνος T_{VCNT} για τις ενεργές γραμμές που θέλουμε να εμφανίσουμε και μετά πρέπει να ακολουθήσει ο παλμός VSYNC (με πράσινο στην εικόνα) ο οποίος είναι active-low για χρονισμό VGA 640 x 480. Για τη δημιουργία του παλμού VSYNC (δηλαδή μετά τις ενεργές γραμμές T_{VCNT}) απαιτείται κάποιος κενός χρόνος πριν τον παλμό T_{VFP} (vertical front porch), κατά τη διάρκεια του παλμού T_{VSP} (vertical sync pulse) και μετά τον παλμό T_{VBP} (vertical back porch). Σε όλη αυτή τη διάρκεια τα pixels είναι «κενά» (πρέπει να έχουν τιμή 0), δεν λαμβάνονται υπόψιν και δεν εμφανίζονται (με γκρι στο κάτω μέρος της εικόνας).

Για ανάλυση (resolution) 640 x 480 με ρυθμό ανανέωσης 60 Hz (60 frames το δευτερόλεπτο) απαιτείται ρολόι 25MHz (περίοδος ρολογιού 40 ns) και οι απαιτούμενοι χρόνοι φαίνονται στους πίνακες παρακάτω:

Horizontal	Clock Cycles	Time (ns)
T_{HCNT}	640	25600
T_{HFP}	16	640
T_{HSP}	96	3840
T_{HBP}	48	1920
Total	800	32000

Vertical	Rows	Clock Cycles	Time (us)
T_{VCNT}	480	384000	15360
T_{VFP}	10	8000	320
T_{VSP}	2	1600	64
T_{VBP}	29	23200	928
Total	521	416800	16672

Πρόσβαση στον κώδικα του εργαστηρίου:

Για τον κώδικα και τις παραδόσεις έχουν δημιουργηθεί git repositories στο gitlab του Τμήματος. Στους εγγεγραμμένους φοιτητές στο εργαστήριο αντιστοιχεί ένα private git repository της μορφής:

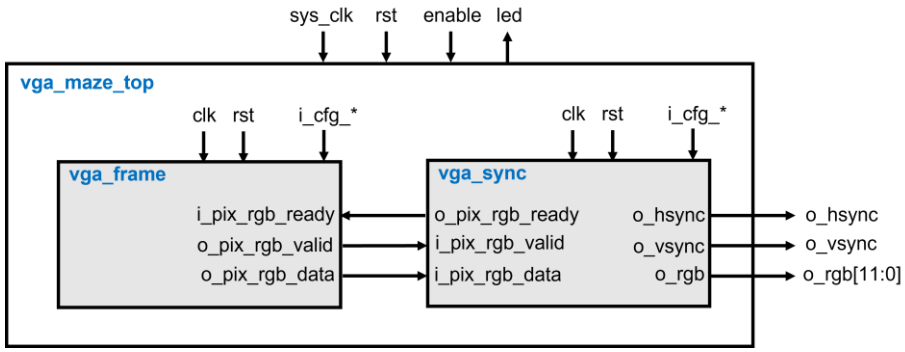
https://gitlab-csd.datacenter.uoc.gr/hy220_2026s/lab1_submits/lab1-csdXYZW

Κάντε `git clone` το προσωπικό σας repository με το username σας (csdXYZW@csd.uoc.gr και το password σας). Η πρόσβαση είναι εφικτή από το δίκτυο του Πανεπιστημίου και το VPN.

ΜΗΝ ΚΑΝΕΤΕ FORK – Εργαστείτε απευθείας στο repository που σας δίνεται

Το σχέδιο του Εργαστηρίου 1:

Για το *Εργαστήριο 1* σας δίνεται το σχέδιο που φαίνεται στην παρακάτω εικόνα:



Το σχέδιο έχει την εξής ιεραρχία:

- **vga_maze_top** *(src/vga_maze_top.sv)*: περιέχει instances από 2 μπλοκ, το **vga_sync** και το **vga_frame** (εξηγούνται παρακάτω), τα οποία συνδέονται όπως φαίνεται στο σχήμα. Επίσης έχει ένα σήμα εισόδου enable που συνδέεται σε διακόπτη (dip switch 0) για την εκκίνηση του συστήματος, ένα σήμα εξόδου led που συνδέεται σε λαμπάκι (led 0), και περιέχει και όλες τις παραμέτρους διαμόρφωσης (configuration) των χρονισμών για ανάλυση 640x480 όπως παρουσιάστηκαν στους πίνακες παραπάνω (*cfg_hcnt, cfg_hfp, cfg_hsp, cfg_hbp, cfg_vcnt, cfg_vfp, cfg_vsp, cfg_vbp*).
- **vga_sync** *(src/vga_sync.sv)*: Το μπλοκ αυτό είναι υπεύθυνο για την υλοποίηση του χρονισμού του πρωτοκόλλου VGA για ανάλυση 640 x 480 και πρέπει να δημιουργεί 2 βασικά σήματα: (1) το HSYNC (horizontal synchronization – οριζόντιος συγχρονισμός) και (2) το VSYNC (vertical synchronization – κατακόρυφος συγχρονισμός) και την έξοδο rgb που είναι οι τιμές για τα χρώματα που πρέπει να εμφανιστούν στην οθόνη (12-bit χρώμα όπου τα 4-MSB είναι κόκκινο/red, τα 4-LSB είναι μπλε/blue και τα 4 μεσαία bits είναι πράσινο/green). Επίσης επικοινωνεί με το μπλοκ **vga_frame** με 3 σήματα (pix_rgb_valid, pix_rgb_ready, pix_rgb_data) που έχουν τη μορφή πρωτοκόλλου χειραψίας valid/ready.

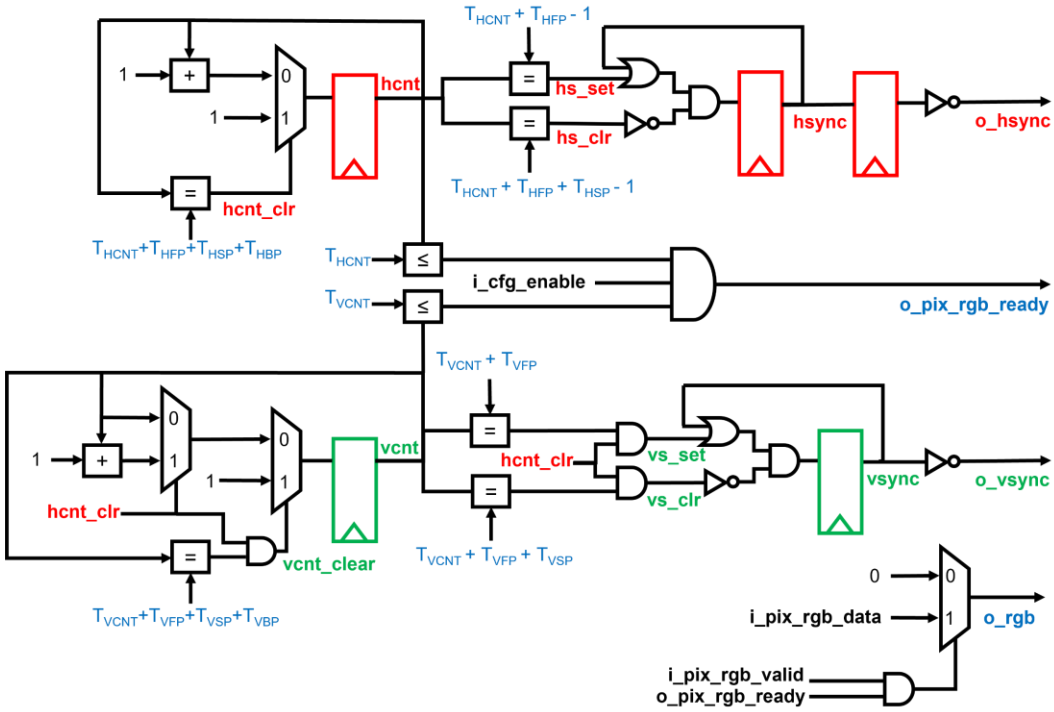
Όποτε το μπλοκ **vga_frame** θέλει να στείλει πληροφορία για τα pixels που πρέπει να εμφανιστούν αναθέτει την τιμή του χρώματος στο pix_rgb_data και ενεργοποιεί το σήμα pix_rgb_valid και περιμένει μέχρι την ενεργοποίηση του pix_rgb_ready. Το μπλοκ **vga_sync** όποτε είναι έτοιμο να δεχτεί τα δεδομένα ενεργοποιεί το σήμα pix_rgb_ready για να σηματοδοτήσει/επιβεβαιώσει ότι δέχτηκε τα δεδομένα pix_rgb_data τα οποία στη συνέχεια θα πρέπει να τα στείλει στην οθόνη. Λεπτομέρειες για το τι θα πρέπει να υλοποιήσετε για αυτό το μπλοκ περιγράφονται παρακάτω.

- **vga_frame** *(src/vga_frame.sv)*: Το μπλοκ αυτό είναι υπεύθυνο για τη δημιουργία των χρωμάτων RGB (red/green/blue) για κάθε pixel που εμφανίζεται στην οθόνη και σας δίνεται έτοιμο. Το μπλοκ δέχεται σαν είσοδο configuration parameters για την ανάλυση (cfg_hcnt, cfg_vcnt) και παράγει σε κάθε κύκλο 12-bit τιμές (pix_rgb_data) που περιέχουν τα τρία χρώματα έτσι ώστε να εμφανίζονται στην οθόνη εναλλάξ κόκκινες, πράσινες, μπλε και μαύρες μπάρες που έχουν ύψος 16 γραμμές. Το μπλοκ αυτό επικοινωνεί με το μπλοκ **vga_sync** με τα σήματα valid/ready που περιγράψαμε παραπάνω και προχωράει στη γέννηση επόμενης τιμής μόνο εφόσον έχει επιβεβαιωθεί η λήψη της προηγούμενης τιμής, δηλαδή έχει προηγηθεί η «χειραψία/handshake» *pix_rgb_valid & pix_rgb_ready*. Στην πλακέτα του εργαστηρίου το κάθε χρώμα είναι 4-bit οπότε μπορούμε να δημιουργήσουμε 2^4 x 2^4 x 2^4 = 4096 διαφορετικά χρώματα. Η τιμή RGB (15, 0, 0) δημιουργεί κόκκινο χρώμα, η τιμή RGB (0, 15, 0) πράσινο χρώμα, η τιμή RGB (0, 0, 15) μπλε χρώμα και η τιμή RGB (0, 0, 0) μαύρο χρώμα. Η τιμή RGB (15, 15, 15) δημιουργεί άσπρο χρώμα και υπάρχουν όλοι οι υπόλοιποι συνδυασμοί που δημιουργούν αποχρώσεις π.χ. το RGB (2, 4, 1) δημιουργεί μια απόχρωση του πράσινου.
- **vga_tb** *(src/vga_tb.sv)*: ένα testbench *για προσομοίωση* που δημιουργεί το ρολόι (25 MHz – 40 ns) και το reset. Το testbench αποθηκεύει την έξοδο του κυκλώματός σας σε ένα αρχείο (vga_log.txt) με το κατάλληλο format έτσι ώστε να μπορείτε να χρησιμοποιήσετε τον VGA Simulator για να βλέπετε τι θα εμφανίζεται στην οθόνη από τον κώδικά σας. Περισσότερες λεπτομέρειες για τον VGA Simulator παρακάτω.
- **Reference output**: Στο φάκελο ref υπάρχει ένα πρότυπο ***vga_log.txt*** output που είναι αυτό που θα πρέπει να δημιουργεί ένας σωστός κώδικας. Μπορείτε να κάνετε diff αυτό που παράγει ο δικός σας κώδικάς με το reference output για να εντοπίσετε λάθη κατά την προσομοίωση. Το format είναι συμβατό με τον VGA Simulator και είναι πολύ απλό. Σε κάθε γραμμή περιέχει το χρόνο σε ns ακολουθούμενο από τις τιμές των σημάτων hsync (1-bit), vsync (1-bit), red (4-bits), green (4-bits), blue (4-bits).
- **VGA Simulator:** Μέσα στο φάκελο ref/vga-simulator υπάρχει μια ιστοσελίδα που μπορείτε να ανοίξετε τοπικά στον web browser σας. Εκεί μπορείτε να επιλέξετε το log file που έχει δημιουργηθεί από την προσομοίωσή σας και όταν πατήσετε το κουμπί submit τότε θα εμφανιστεί σε μια εικονική VGA οθόνη η έξοδός σας. Μπορείτε να το δοκιμάσετε επίσης με το reference output. Μην αλλάξετε τις παραμέτρους που υπάρχουν ήδη στη σελίδα! Credits: Ο VGA Simulator έχει δημιουργηθεί από τον Eric Eastwood στο παρακάτω website http://ericeastwood.com/lab/vga-simulator/
- Δίνεται επίσης ο φάκελος **run** με έτοιμο Makefile για να τρέξετε την προσομοίωση και να δείτε αν βγάζετε σωστά αποτελέσματα (make και μετά make check) με τους υποστηριζόμενους προσομοιωτές όπως αναφέρονται στη σελίδα του μαθήματος.

Τι πρέπει υλοποιήσετε και να προσομοιώσετε πριν πάτε στα εργαστήρια:

Για το εργαστήριο αυτό θα πρέπει να υλοποιήσετε το μπλοκ **vga_sync** του οποίου ένα άδειο module σας δίνεται. Με βάση τις προδιαγραφές του χρονισμού VGA 640 x 480 που παρουσιάστηκαν παραπάνω σας δίνεται το παρακάτω σχηματικό με πύλες που υλοποιεί τις προδιαγραφές. Περιλαμβάνει 2 μετρητές τον hcnt και τον vcnt που μετρούν τις στήλες και τις

γραμμές αντίστοιχα και οι οποίοι πρέπει να γίνονται reset στην τιμή 1. Μετά τους μετρητές υπάρχουν set-clear flip-flops για να δημιουργηθούν κατάλληλα οι παλμοί HSYNC και VSYNC. Επίσης πρέπει να δημιουργείται το σήμα *o_pix_rgb_ready* που πηγαίνει στο μπλοκ **vga_frame** (σας δίνεται έτοιμο). Το σήμα *o_pix_rgb_ready* πρέπει να είναι ενεργό μόνο τους κύκλους που θα παρέχουμε στην οθόνη ενεργά pixels και οι τιμές RGB πρέπει να προέρχονται από την είσοδο *i_pix_rgb_data* διαφορετικά πρέπει να είναι «κενά» (πρέπει να έχουν τιμή 0). Για τον παλμό HSYNC έχει προστεθεί ένας κύκλος καθυστέρηση έτσι ώστε το μπλοκ **vga_frame** να βγάλει την έξοδό του (pixel) συγχρονισμένα με το μπλοκ **vga_sync**.



Θα πρέπει να υλοποιήσετε σε SystemVerilog RTL το κύκλωμα για το μπλοκ **vga_sync**. Πριν πάτε στο εργαστήριο θα πρέπει να προσομοιώσετε και να επαληθεύσετε με το έτοιμο testbench και τα reference outputs τον κώδικά σας. Θα πρέπει να μπορείτε να δείτε την έξοδο που θα έβγαζε το κύκλωμα με τον προσομοιωτή VGA Simulator. Μη ξεχάσετε να βάλετε reset στους καταχωρητές!

Τι πρέπει να κάνετε στο εργαστήριο:

Θα πρέπει να πάτε στο εργαστήριο με τον **κώδικά σας έτοιμο και προσομοιωμένο από πριν** και να ακολουθήσετε την ροή του εργαλείου Xilinx Vivado και τα βήματα που χρειάζεται για να «κατεβάσετε» το σχέδιο στην FPGA και να το δείτε να δουλεύει. Δείξτε το κύκλωμα που δουλεύει στο βοηθό.

Σημείωση: Το ρολόι του κυκλώματος πρέπει να είναι 25MHz (περίοδος ρολογιού 40ns) για τη σωστή υλοποίηση του χρονισμού VGA με ανάλυση 640x480 (resolution) και ρυθμό ανανέωσης 60Hz (refresh rate), δηλαδή 60 VGA frames per second. Η πλακέτα του εργαστηρίου όμως έχει εξωτερικό ρολόι 100 MHz οπότε υπάρχει ένα επιπλέον block από τον «Clocking Wizard» του Vivado *(src/clk_wiz.sv)* το οποίο είναι ήδη συνδεδεμένο στο top και ενεργοποιείται μόνο κατά την σύνθεση. Για την ανάθεση pins (pin assignment) χρησιμοποιείστε το constraint file που σας δίνεται (constr/lab1.xdc).

Παράδοση του κώδικα του εργαστηρίου:

Πρέπει να παραδώσετε στο τέλος του εργαστηρίου τα αρχεία στον **φάκελο src** όπως έχουν προκύψει/αλλάξει. Η παράδοση θα πρέπει να γίνει **μέχρι τη 2η εβδομάδα των εργαστηρίων (αναλόγως το Γκρουπ που ανήκετε)** στο τέλος της ώρας που έχετε εργαστήριο **και πρέπει να έχετε εξεταστεί από τον βοηθό** – δεν αρκεί μόνο να παραδώσετε στο gitlab!

Οι παραδόσεις θα γίνονται μόνο μέσω του gitlab.
ΜΗΝ προσθέσετε το Vivado project ή το φάκελο "run" στο git !!!

Ενδεικτικές εντολές

- git status (για να δείτε τι έχει αλλάξει)
- git config user.email csdXYZW@csd.uoc.gr (ΜΗΝ χρησιμοποιήσετε το --global flag)
- git config user.name "Name Surname" (ΜΗΝ χρησιμοποιήσετε το --global flag)
- git add src (για να προσθέσετε τα αρχεία που αλλάξατε στο φάκελο src)
- git commit -m "lab1 delivery" (commit και μήνυμα)
- git push (θα πρέπει πάλι να χρησιμοποιήσετε το username και το password σας)
- **Επιβεβαιώστε ότι το push έγινε και ότι τα αρχεία έχουν ανέβει στο gitlab στο repository που σας δόθηκε και ΟΧΙ ΣΕ FORK**

Μετά την πάροδο της ημερομηνίας του Εργαστηρίου σας
ΔΕΝ θα μπορείτε να παραδώσετε και να κάνετε push στο gitlab !